Учреждение образования

«Белорусский государственный технологический университет»

**Кафедра информационных систем и технологий**

**«Отчёт по лабораторной работе 7»**

“ ИССЛЕДОВАНИЕ ПОТОКОВЫХ ШИФРОВ”

**Выполнил:** студент 3 курса

4 группы специальности ПОИТ

Примаков Максим Николаевич

**Проверил:** преподаватель

Блинова Евгения Александровна

Минск 2021

Потоковый шифр (иногда говорят «поточный») – симметричный шифр, преобразующий каждый символ mi открытого текста в символ шифрованного, ci, зависящий от ключа и расположения символа в тексте. Термин «потоковый шифр» обычно используется в том случае, когда шифруемые символы открытого текста представляются одной буквой, битом или реже – байтом. Все потоковые шифры делятся на 2 класса: синхронные и асинхронные (или самосинхронизирующиеся).

Синхронные потоковые шифры (СПШ) — шифры, в которых поток ключей генерируется независимо от открытого текста и шифротекста.

Самосинхронизирующиеся потоковые шифры (асинхронные потоковые шифры (АПШ)) — шифры, в которых ключевой поток создаётся функцией ключа и фиксированного числа знаков шифротекста.

Основной задачей потоковых шифров является выработка некоторой последовательности (гаммы) для зашифрования, т.е. выходная гамма является ключевым потоком (ключом) для сообщения.

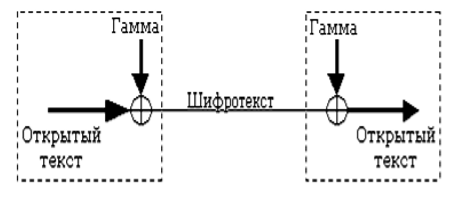


Рисунок 1.1 – Схема потокового шифра

Случайные числа (последовательности) – последовательность элементов, каждый из которых не может быть предсказан (вычислен) только на основе знания предшествующих ему элементов данной последовательности.

Псевдослучайные числа – последовательность элементов, полученная в результате выполнения некоторого алгоритма и используемая в конкретном случае вместо последовательности случайных чисел.

Алгоритм **линейного конгруэнтного генератора**, описываетя следующим рекуррентным соотношением:

xt+1 = (a\*xt + c) mod n.

а, с и n – константы. Период такого генератора (период ПСП) не превышает n.

**Генератор ПСП на основе регистров сдвига.** РС с линейной обратной связью (РСЛОС) состоит из двух частей: собственно РС и функции обратной связи. На рис. 6.2 представлена общая схема РС с линейной обратной связью. Функция обратной связи реализуется с помощью сумматоров сложения по модулю два (элементы XOR; на рис. 6.2 обозначены в виде кружочков со знаком сложения).

РСЛОС строятся на основе примитивных порождающих полиномов (многочленов), которые мы подробно анализировали при изучении циклических помехоустойчивых кодов. Если многочлен является неприводимым, то период ПСП при ненулевом начальном условии (ненулевом состоянии) регистра будет максимально возможным: 2L – 1.

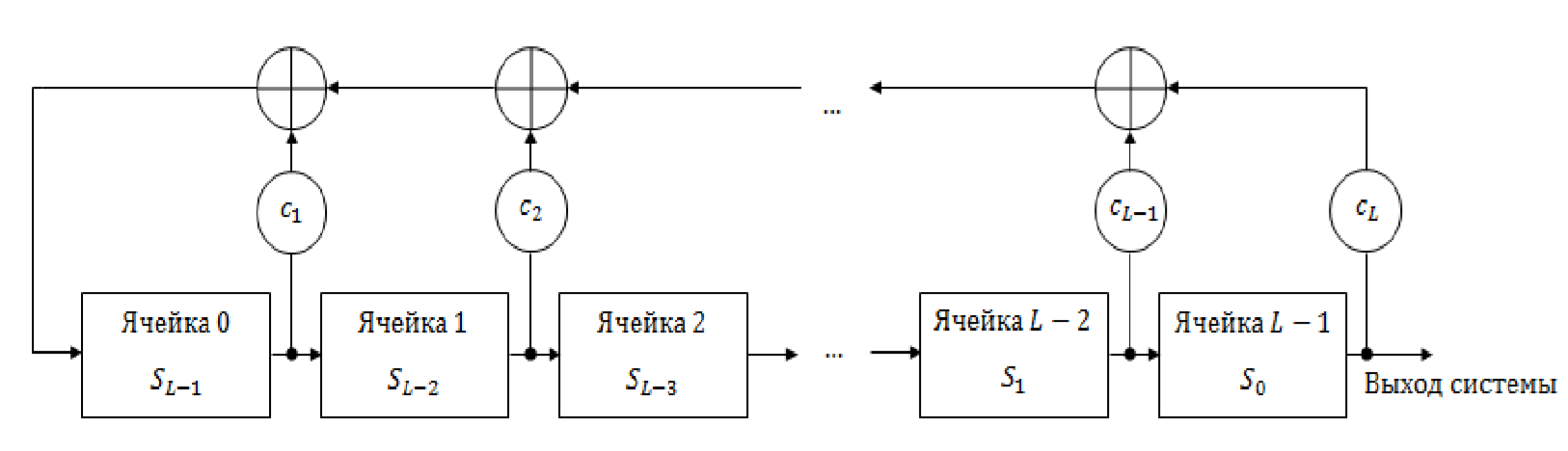


Рисунок 1.2 – Общая схема регистра сдвига с линейной обратной связью

Генератор псевдослучайных чисел на основе **алгоритма RSA** разработан для систем асимметричного зашифрования/расшифрования. Генератор ПСП на основе RSA устроен следующим образом. Последовательность генерируется с использованием соотношения:

xt = (xt-1)е mod n.

Начальными параметрами служат n, большие простые числа p и q (причем n = p\*q), целое число е, взаимно простое с произведением (р – 1)\*( q –1), а также некоторое случайное начальное значение, x0. Выходом генератора является на t-м шаге является младший бит числа xt. Безопасность генератора опирается на сложности взлома алгоритма RSA, т. е. на разложении числа n на простые сомножители.

**Алгоритм BBS** или генератором на основе квадратичных вычетов. Начальное значение x0 генератора вычисляется на основе соотношения:

x0= x2 mod n

где n, как и в генераторе на основе RSA, является произведением простых чисел p и q, однако в нашем случае эти простые числа должны быть сравнимы с числом 3 по модулю 4, т. е. при делении p и q на 4 должен получаться одинаковый остаток: 3; число x должно быть взаимно простым с n; число n называют числом Блюма. Выходом генератора на t-м шаге является младший бит числа xt:

xt = (xt-1)2 mod n.

**Алгоритм RC4**, как и любой потоковый шифр, строится на основе генератора псевдослучайных битов (генератора ПСП). На вход генератора записывается ключ, а на выходе читаются псевдослучайные биты. Длина ключа может составлять от 40 до 2048 бит.

Ядро алгоритма состоит из функции генерации ключевого потока. Другая часть алгоритма – функция инициализации, которая использует ключ переменной длины Ki для создания начального состояния генератора ключевого потока.

В основе алгоритма – размер блока или слова, определяемый параметром n. Обычно n = 8, но можно использовать и другие значения. Внутренне состояние шифра определяется массивом слов (S-блоком) размером 2n. При n = 8 элементы блока представляют собой перестановку чисел от 0 до 255.

**ЗАДАНИЕ 1.** Генерация ПСП используя алгоритм Линейного конгруэнтного генератора.

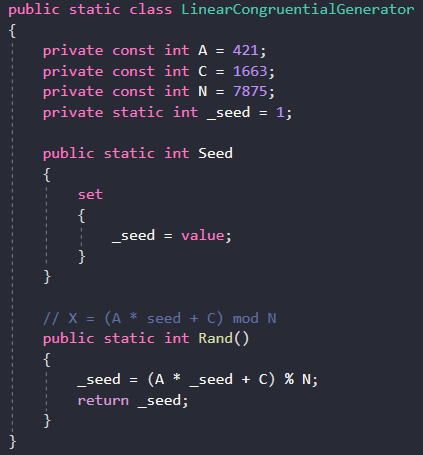
Часто используемый алгоритм генерирования (программно или аппаратно) ПСП реализуется на основе так называемого линейного конгруэнтного генератора, описываемого следующим рекуррентным соотношением:



где: xt и xt+1 – соответственно t-й (предыдущий) и (t+1)-й (текущий, вычисляемый) члены числовой последовательности; а, с и n – константы. Период такого генератора (период ПСП) не превышает n.

Для работы с этим алгоритмом, потребуются некоторые входные параметры:

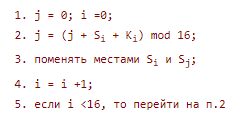
Код реализации на языке C#:



**Задание 2:**

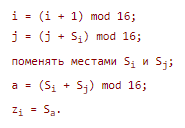
*Алгоритм* *RC4* состоит из двух этапов. На первом, подготовительном этапе производится *инициализация* таблицы замен S. На втором, основном этапе вычисляются псевдослучайные числа.

Посмотрим, как инициализируется *таблица* S. Вначале она заполняется последовательно числами от 0 до 15. *Ключ* представляется в виде последовательности 4-битовых слов, которыми заполняется другой *массив* K, такого же размера, как S. Если *ключ* оказался короче, чем надо, он повторяется нужное число раз. Затем выполняются следующие действия ( *алгоритм 1* ):



В результате выполнения этого алгоритма производится начальное заполнение таблицы замен S, причем это начальное перемешивание значений производится в зависимости от секретного ключа.

После того, как *таблица* S подготовлена, можно начинать генерацию случайных n-битовых слов. Для этого счетчикам i и j присваивается начальное *значение* 0. Затем для получения каждого нового значения zi выполняются следующие действия ( *алгоритм 2* ):

****

Код реализации на языке C#:

